WebGL - TP Noté

Pierre BIASUTTI
IUT Informatique, Licence Pro

1 Visualisateur 3D

1.1 Idée générale

Ce TP à pour but de réaliser un visualisateur 3D intéractif. Ce visualisateur devra afficher un cube et remplir certaines fonctionnalités obligatoires, ainsi que d'autres fonctionnalités facultatives. Un exemple du rendu final est illustré Figure 1. Le TP est organisé selon une difficulté incrémentale et doit être réalisé dans l'ordre des questions. Toutes les parties sont obligatoires exceptées celles suivies de la mention "facultatif".

A noter que le code rendu doit être correctement structuré et ne doit pas présenter de redondances. D'autre part, n'hésitez pas à réutiliser le code que vous avez produit pendant les TPs, notamment les shaders.

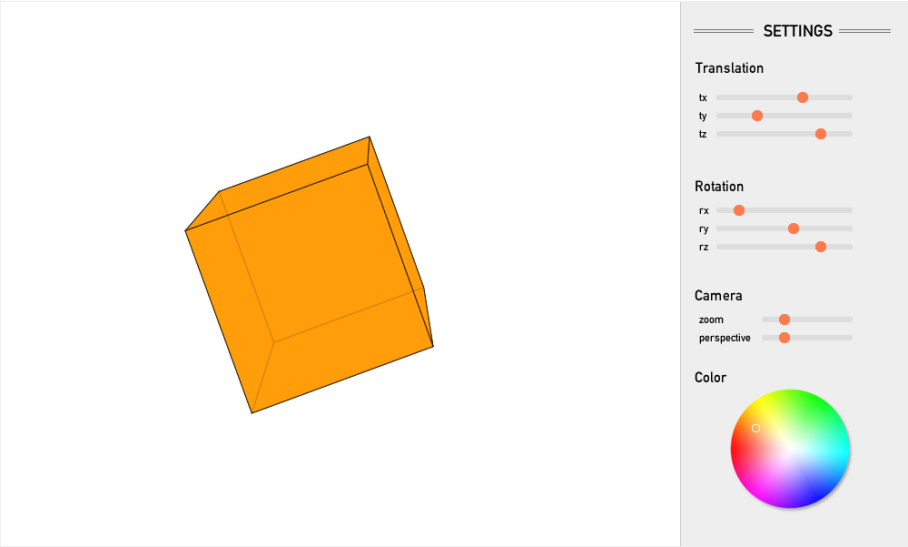

Figure 1: Schéma d'illustration du visualisateur 3D.

1.2 Bases du visualisateur

Création du cube La première partie consiste à afficher un cube en 3D dans le canvas de la page. Cette partie correspond directement à l'exercice 3 du TP 3 du cours (`http://www.labri.fr/perso/pbiasutt/Cours/WebGL/TP/3/tp3.pdf`). La position des sommets formant le cube, ainsi que la couleur associée à chaque sommet du cube doivent être stockées en utilisant deux buffers. Chaque face du cube doit être d'une couleur différente.

Perspective et Transformations Tout comme dans les TPs, on utilisera la bibliothèque "gl-matrix" (`http://www.labri.fr/perso/pbiasutt/Cours/WebGL/TP/3/gl-matrix-min.js`) pour gérer les transformations (perspective, translation rotation). Modifiez votre vertex shaders pour inclure des matrices 4x4 de perspective, de rotation et de translation (exactement comme dans le TP 3, exercice 1 et 2).

1.3 Intéractions avec sliders

Tout comme dans la Figure 1, on souhaite disposer de 8 sliders (`https://www.w3schools.com/howto/howto_js_rangeslider.asp`) pour contrôler la scène. Lorsque que l'on change la valeur d'un slider, la modification doit se faire directement dans le canvas. Ces sliders doivent gérer:

1. Translation sur l'axe x
2. Translation sur l'axe y
3. Translation sur l'axe z
4. Rotation autour de l'axe x
5. Rotation autour de l'axe y
6. Rotation autour de l'axe z
7. Zoom
8. Champ de vision (yFov dans la matrice de perspective)

Les valeurs des sliders pour la translation doivent varier entre -10 et 10. Les valeurs des sliders pour la rotation doivent varier entre -PI et PI. La valeur du slider pour le zoom doit varier entre 0.1 et 5. Enfin le champ de vision doit pouvoir varier entre 30 et 120 degrés.

1.4 Couleurs (facultatif)

Rajoutez un "colorpicker" Javascript de sorte à pouvoir contrôler la couleur du cube. Le "colorpicker" est un élément permettant de choisir une couleur sur une roue chromatique. A partir de cette couleur, il faudra modifier le buffer de couleur de sorte à changer la couleur du cube. Attention cependant: si toutes les faces sont de la même couleurs, le cube sera difficile à voir car la scène ne possède pas d'éclairage. Le cube apparaitra donc d'une couleur uniforme. Pour palier ce problème, vous pouvez altérer légèrement la couleur issue du colorpicker pour chaque face. On peut par exemple obtenir une couleur proche, mais différente, d'une couleur r,g,b en définissant une nouvelle couleur r+0.1, g, b (idem sur g et sur b).

1.5 Souris (facultatif)

On cherche maintenant à pouvoir intéragir avec la souris directement dans le canvas. Pour exemple, le clic gauche, lorsqu'il est maintenu pendant un déplacement sur le canvas entraine une rotation. De plus, le scroll peut faire varier le zoom. Vous pouvez implémenter ces interactions en utilisant les évenements Javascript ou jQuery.

2 Rendu du TP Noté

Tous les fichiers utilisés dans le TP noté doivent être ajoutés à un dépôt GIT. Vous devez envoyer un email avec l'url du dépôt GIT au chargé de TD avant la deadline du TP noté. Cet email doit avoir pour objet : [WebGL] Rendu TP Noté - Nom Prénom.

Bonne chance !